Human-like Planning of Swerve Maneuvers for Autonomous Vehicles

Tianyu Gu[1], John M. Dolan[2] and Jin-Woo Lee[3]

Abstract— **In this paper, we develop a motion planner for on-road autonomous swerve maneuvers that is capable of learning passengers' individual driving styles. It uses a hybrid planning approach that combines sampling-based graph search and vehicle model-based evaluation to obtain a smooth trajectory plan. To automate the parameter tuning process, as well as to reflect individual driving styles, we further adapt inverse reinforcement learning techniques to distill human driving patterns from maneuver demonstrations collected from different individuals. We found that the proposed swerve planner and its learning routine can approximate a good variety of maneuver demonstrations. However, due to the underlying stochastic nature of human driving, more data are needed in order to obtain a more generative swerve model.**

I. INTRODUCTION

In the last two decades, developments in vehicle autonomy technologies have brought societal attention to research in autonomous passenger vehicles (APV). Motion planning (MP) is a key component. It generates prescribed motion trajectories to navigate an APV, preferably in a smooth and human-like fashion.

The core of existing MP techniques usually involves a single optimization routine based on manually tuned reward/cost functionals. While some planning tasks have non-ambiguous optimality criteria, such as (minimum) time in planners for race cars, there is no single universally acknowledged criterion that defines what is "good" for general on-road driving. One reason is that the driving quality should be at least in part based on human preferences, which obviously can vary from person to person. Two challenges arise: first, individual driving style should be taken into account explicitly in the tuning procedure; second, manual tuning for this purpose can be very difficult and time-consuming.

In this paper, we address these challenges by developing a parameterized motion planner for swerve maneuvers and utilizing inverse reinforcement learning to recover the optimal parameterization to best mimic human demonstrations.

II. RELATED WORK

Trajectory planners for on-road autonomous driving can be viewed as short-ranged or long-ranged. Short-range (local) trajectory planning algorithms have been developed for autonomous driving. Lee [1] proposed a real-time local lane-following and lane-change trajectory generation routine. The winning entries in the DARPA Grand and Urban Challenges, "Stanley" [2] and "Boss" [3], used a planning scheme based on local trajectory sampling to follow the lane centerline. Long-range planners use graph-based or optimization-based procedures to take a longer preview horizon into account, and behave more robustly in complicated driving scenarios. Ziegler [4] and McNaughton [5] used the spatiotemporal state-lattice planning scheme for on-road driving. Ziegler [6] proposed constraint-based trajectory generation by continuous optimization routines that optimize certain optimal criterion. Li [7] developed an on-road planner by applying support vector machine (SVM) in the corridor environment.

The optimality criteria of these prior planners and many other motion planners developed for other mobile robots or ground vehicles [8], [9], [10], [11] is typically defined by cumulative weighted feature (cost) terms to account for different aspects of driving. The weights are often obtained through a manual tuning process, which is often very difficult and time-consuming. Meanwhile, the individual driving style was not taken into account when performing the weight tuning.

Learning-from-demonstration techniques provide methods for automatic tuning in a variety of planning/control applications. Hamner [12] developed a potential field-based controller for obstacle avoidance, then applied and compared several parameter fitting routines in order to mimic a few demonstrated avoidance paths. Abbeel [13] applied apprenticeship to a fused search/optimization-based path planner to learn human parking styles. Silver [14] formulated the learning as an unconstrained optimization problem and applied it to a robot exploring in complex and unstructured terrain. These methods all developed certain path planner, and depend on iterative optimization techniques to automatically find the weights for their cost (feature) terms. For on-road driving, on the other hand, the learning should be applied to the trajectory planner, which generates not only a path plan, but also speed information.

In Gu [15], we proposed a trajectory planner and a learning routine in order to mimic a human-like driving style affected by road geometry, yet not capable of reacting to the surrounding objects. The contribution of this paper is the development of a trajectory planner and an automated parameter tuning (learning) scheme for object swerve avoidance maneuver to discover individual driving styles. To the best of our knowledge, there has been no prior work on learning driving styles for on-road swerve maneuver of an APV.

The remainder of the paper is organized as follows.

[1] Tianyu Gu, *IEEE Student Member*, is with Department of Electrical & Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA Tianyu@cmu.edu

[2] John M. Dolan is with the Robotics Institute and the Department of Electrical & Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA jmd@cs.cmu.edu

[3] Jin-Woo Lee is with the Research & Development, General Motors, Warren, MI, USA jin-woo.lee@gm.com

Section III describes the proposed parameterized planning scheme for swerve maneuvers. Section IV explains the learning techniques for parameter auto-tuning in order to distill individual driving preference from demonstrations. Sections V and VI conclude with experimental results and discussion.

III. PLANNING FOR SWERVE MANEUVER

The proposed swerve planner consists of two planning components: a sampling-based path planner to generate a coarse spatial curve, and a vehicle model-based trajectory generator to obtain a smooth and kinematically feasible trajectory that is collision-free to surrounding objects.

A. Heuristic Sampling-based Path Planning

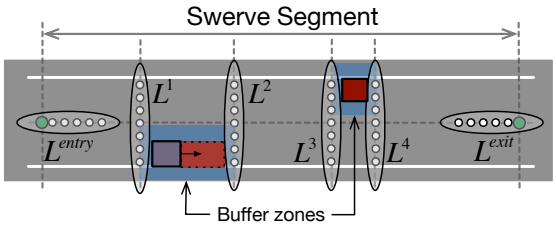


Fig. 1: Sampling-based heuristics for path planning. A swerve segment consisting of two consecutive swerve maneuvers is illustrated. The buffer zones around the objects are represented by blue regions. Layers of waypoints are placed longitudinally and laterally. Longitudinal sampled layers (L^{entry}, L^{exit}) are the entry and exit of a swerve segment, each of which consists of a single (green) node marking the start or end. Lateral sampled layers (L^1, L^2, L^3, L^4) are aligned to the front and back edges of the objects' buffer zones. Piecewise-linear paths are created by generating all possible edges between the nodes in neighboring layers.

A directed graph is constructed for planning using the following state sampling heuristics: a swerve segment (Fig. 1) is defined as layers of sampled waypoints (nodes). Two special single-node layers are placed at the entry L^{entry} and exit L^{exit} of the swerve segment. The edges (piecewise-linear paths) are obtained by connecting layers of nodes. An edge e connects nodes n_e^i from layer i and n_e^{i+1} from layer $i+1$. Each node n has coordinates $[s(n), l(n)]$ in the road frame, where functions s and l return the longitudinal and lateral positions of the node n. Each edge is evaluated with a cost, which is the linear combination of weighted feature terms:

$$\Omega^T \hat{\Phi}(e) = \sum_{m=1}^{M} \omega_m \cdot \hat{\phi}_m(e) \tag{1}$$

where M is the dimension of the feature space, Ω is the weight vector, and $\hat{\Phi}$ is the normalized feature vector of the original features of different scales, such that $\hat{\phi}_i \in [0, 1]$:

$$\Omega = [\omega_1, ..., \omega_M]^T$$
$$\hat{\Phi} = [\hat{\phi}_1, ..., \hat{\phi}_M]^T = [\frac{\phi_1}{max(\phi_1)}, ..., \frac{\phi_M}{max(\phi_M)}]^T$$

To capture the core factors of a swerve maneuver, three categories of five feature terms and the corresponding weight parameters are defined: ϕ_{offset} penalizes offset from the original centerline by the area of deviation; ϕ_{swerve} punishes excessive swerves with high latitude/station displacement ratio (note that three different feature terms are derived based on the nature of an edge, i.e., whether it is deviating from the centerline, getting back to, or otherwise), and $\phi_{obstacle}$ penalizes paths close to obstacles with an exponential decay as the path gets farther away.

$$\phi_{offset}(e) = (|l(n_e^{i+1})| + |l(n_e^i)|)(|s(n_e^{i+1})| - |s(n_e^i)|)$$

$$\phi_{obstacle}(e) = \begin{cases} 0, & e \cap (Z_{buffer} \cup Z_{fatal}) = \emptyset \\ e^{-d^{obj}(e)}, & e \cap Z_{buffer} \neq \emptyset \ \& \ e \cap Z_{fatal} = \emptyset \\ \infty, & e \cap Z_{fatal} \neq \emptyset \end{cases}$$

$$\phi_{swerve}(e) = \frac{|l(n_e^{i+1})| - |l(n_e^i)|}{|s(n_e^{i+1})| - |s(n_e^i)|)}$$
$$= \begin{cases} \phi_{swerve}^{outside}, & l(n_e^{i+1}) \cdot l(n_e^i) \geq 0 \ \& \ |l(n_e^{i+1})| > |l(n_e^i)| \\ \phi_{swerve}^{inside}, & l(n_e^{i+1}) \cdot l(n_e^i) \geq 0 \ \& \ |l(n_e^{i+1})| < |l(n_e^i)| \\ \phi_{swerve}^{other}, & \text{otherwise} \end{cases}$$

where Z_{fatal} and Z_{buffer} are the fatal (collision) and buffer zones regarding an object, and the function d^{obj} returns the distance of an input edge to its nearest obstacle.

On the directed acyclic graph (DAG) we built, the planner solves a deterministic Markov Decision Process (dMDP) problem to find the optimal state transition sequence $\mathbf{e}^*$:

$$\mathbf{e}^* = \arg\min_{\mathbf{e}} \sum_{e \in \mathbf{e}} \Omega^T \hat{\Phi}(e) \tag{2}$$

where $\mathbf{e}^*$ is the optimal sequence of state transitions from the start node n^{entry} to end node n^{exit}. Therefore, $\mathbf{e}^*$ represents a piecewise-linear path $\boldsymbol{P}_{linear}$ to be further smoothed.

B. Model-based Trajectory Generation

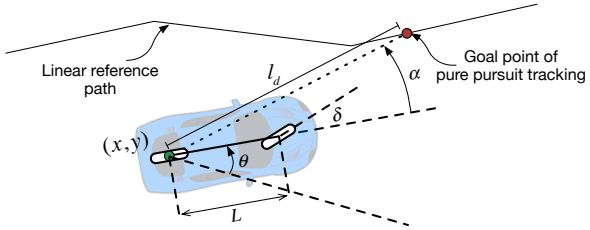


Fig. 2: The path smoothing component of the model-based trajectory generation is based on a pure-pursuit tracking controller and a kinematic bicycle vehicle model.

For trajectory generation, a smooth path is obtained by evaluating $\boldsymbol{P}_{smooth}$ with a vehicle model (Fig. 2). The model consists of a kinematic half-car model (3) and a pure pursuit controller (4). The evaluation process guarantees the kinematic feasibility of the smoothed path.

$$\begin{aligned} \dot{x} &= v \cdot cos(\theta) \\ \dot{y} &= v \cdot sin(\theta) \\ \dot{\theta} &= v \cdot \kappa \\ \dot{\kappa} &= \frac{tan(\delta)}{L} \end{aligned} \tag{3}$$

$$\delta(l_d) = tan^{-1}(\frac{2Lsin(\alpha)}{l_d}) \tag{4}$$

The look-ahead parameter l_d of the pure-pursuit controller determines the smoothness of the evaluated path. The larger l_d is, the smoother $\boldsymbol{P}_{smooth}$ is, but it will deviate more from $\boldsymbol{P}_{linear}$ (the effect of corner cutting) with a non-straight path sequence. Note that l_d is a pre-determined value, which is not included in the following parameters for the path planner:

$$\Omega = [\omega_{offset}, \omega_{swerve}^{outside}, \omega_{swerve}^{inside}, \omega_{swerve}^{other}, \omega_{obstacle}]^T \quad (5)$$

For APV, lateral and longitudinal accelerations a_{lat} and a_{lon} are crucially important for user comfort. Under the kinematic model assumption (i.e., moving along the path without skidding) along path $\boldsymbol{P}_{smooth}$, which contains a sequence of path points $\{\boldsymbol{p}_i\}$, the following relations hold:

$$\begin{aligned} a_{lat} &= \kappa \cdot v_{lon}^2 \\ a_{lon} &= \dot{v}_{lon} \end{aligned} \quad (6)$$

where κ and v_{lon} are the curvature value and the longitudinal speed of a point $\boldsymbol{p}$ on the smooth path $\boldsymbol{P}_{smooth}$.

In order to further augment a geometric path with temporal information, a speed profile $\boldsymbol{V}_{lon}$ can be generated such that each point $\boldsymbol{p}$ on the reference path $\boldsymbol{P}_{smooth}$ has a corresponding speed value $v_{lon}(\boldsymbol{p})$ subject to the following dynamics constraints:

$$\begin{aligned} v_{lon}(\boldsymbol{p}) &\leq \sqrt{\frac{a_{lat}}{|\kappa(\boldsymbol{p})|}} \\ \dot{v}_{lon} &\leq a_{lon}^{+} \\ \dot{v}_{lon} &\geq -a_{lon}^{-} \end{aligned} \quad (7)$$

where a_{lat} is the lateral acceleration maximum, and a_{lon}^{+} / a_{lon}^{-} are the positive/negative longitudinal acceleration maximum. A constraining routine [15] is used to generate a first-order continuous speed profile. The speed generator can therefore be completely parameterized by:

$$\Lambda = [a_{lat}, a_{lon}^{+}, a_{lon}^{-}]^T \quad (8)$$

The other *external*[1] comes from collision status w.r.t. the surrounding objects. Given smooth path $\boldsymbol{P}_{smooth}(\Omega')$ and speed profile $\boldsymbol{V}_{lon}(\Lambda')$, a spatiotemporal trajectory $\boldsymbol{\mathcal{T}}(\Omega', \Lambda')$ can be used to calculate for collision checking against both static and moving objects. When a collision is detected, the pair of parameters Ω' and Λ' that generate trajectory $\boldsymbol{\mathcal{T}}$ will be marked as invalid parameterization.

IV. LEARNING DRIVING STYLE

Parameter tuning can be viewed as a learning process in order to mimic the human driving pattern, which includes the following aspects of the swerve maneuver:

1) When and how to start and exit a swerve?
2) How far to keep from a particular type of obstacle?
3) How fast to drive during a swerve?

Typically, learning by empirical manual tuning has no systematic approach. An algorithm designer would have to perform on-line tuning according to some rules of thumb and with subjective feedback (most often from the designer himself). Such a trial-and-error approach is difficult and labor-intensive. It is therefore ideal to automate the parameter-tuning process to offload this burden.

Notice that the proposed planner has two sequential components, which have been parameterized by Ω and Λ. We formulate an inverse reinforcement learning procedure, whose goal is to recover optimal parameters Ω^* and Λ^* that best replicate demonstration trajectories, such that:

$$\begin{aligned} \Omega^* &= \underset{\Omega \in \arg\min_{\Omega} \epsilon_{path}}{\arg\min} \|\Omega\|_2 \text{ s.t. } \sum \Omega = 1 \text{ and } \Omega \geq \mathbf{0} \\ \Lambda^* &= \underset{\Lambda}{\arg\min}\ \epsilon_{speed} \text{ s.t. } \Lambda \geq \mathbf{0} \end{aligned} \quad (9)$$

where $\sum \Omega$ represents the element-wise sum of vector Ω, ϵ_{path} and ϵ_{speed} are the cumulative errors (the root of the average least-square path/speed deviations) between the model-evaluated trajectory and demonstration trajectory:

$$\epsilon_{path} = \sqrt{\sum_{i=1}^{N} |\boldsymbol{p}^i - \tilde{\boldsymbol{p}}^i|^2 / N} \quad \epsilon_{speed} = \sqrt{\sum_{i=1}^{N} |v_{lon}^i - \tilde{v}^i|^2 / N}$$

where $\forall \boldsymbol{p}^i \in \boldsymbol{P}_{smooth}, \forall \tilde{\boldsymbol{p}}^i \in \tilde{\boldsymbol{P}}, \forall v_{lon}^i \in \boldsymbol{V}_{lon}, \forall \tilde{v}^i \in \tilde{\boldsymbol{V}}$, $\tilde{\boldsymbol{P}}$ and $\tilde{\boldsymbol{V}}$ are the path and speed profiles of the demonstration trajectory.

The reason for choosing Ω^* that minimizes $\|\Omega\|_2$ is that there is typically more than one configuration of Ω that achieves the same dMDP traversal sequence in path planning, therefore the same $\boldsymbol{P}_{smooth}(\Omega)$ and the same ϵ_{path}. We use the same conditioning technique used in Abbeel [13] to find the most uniformly parameterized Ω that achieves this resolution-complete optimality. The overall algorithm is outlined in Fig. 3.

Step-1: Obtain solution of MDP planning problem given the parameter vector Ω

$$\mathbf{e}^* = \arg\min_{\mathbf{e}} \sum_{e \in \mathbf{e}} \Omega^T \hat{\Phi}(e)$$

Step-2: Construct a coarse path $\boldsymbol{P}_{coarse}$ by connecting the piecewise-linear edges in $\mathbf{e}^*$.
Step-3: Apply the trajectory generation model to $\boldsymbol{P}_{coarse}$ given parameter vector Λ to obtain $\{\boldsymbol{P}_{smooth}, \boldsymbol{V}_{lon}\}$
Step-4: Evaluate path optimization criterion by calculating cumulative point-wise weighted errors of lateral offsets to the original reference $\tilde{\boldsymbol{P}}$:

$$E_{path} = \sum_{\boldsymbol{p}^i \in \boldsymbol{P}_{smooth}, \tilde{\boldsymbol{p}}^i \in \tilde{\boldsymbol{P}}} \frac{l(\tilde{\boldsymbol{p}}^i)}{max(l(\tilde{\boldsymbol{p}}^i))} \|l(\boldsymbol{p}^i) - l(\tilde{\boldsymbol{p}}^i)\|^2$$

Step-5: (After determination of Ω^*) evaluate speed profile error against speed demonstration $\tilde{\boldsymbol{V}}$:

$$E_{speed} = \sum_{v_{lon}^i \in \boldsymbol{V}_{lon}, \tilde{v}^i \in \tilde{\boldsymbol{V}}} \|v_{lon}^i - \tilde{v}^i\|^2$$

Fig. 3: Planner forward evaluation routine for inverse reinforcement learning. The parameters for path planning and model-based trajectory generation are learned sequentially for efficiency.

[1] As it is the constraints imposed by surrounding objects, rather than the dynamics of the trajectory itself.

The learning process requires careful choice of optimization routines, particularly for the path parameter Ω learning subroutine due to its high likelihood of local optimality in the dMDP formulation. Probabilistic global optimization methods, i.e., simulated annealing, are suitable. Learning the parameter Λ of the speed model can be efficiently done with the Nelder-Mead algorithm.

V. RESULTS

A. Data Collection

To collect real human driving data for experiments, we mount a centimeter-level real-time kinematic (RTK) GPS receiver on the roof of the vehicle, located at the center of the rear axle, for data collection (Fig. 4). The driving environment was a single-lane corridor in a closed course, and the locations of the objects were surveyed a priori with the same level of accuracy.

(a) **(b)**

Fig. 4: Data collection (a) vehicle setup with GPS antenna mounted on the roof of the car. (b) test scenario setup where orange cones were placed partially blocking the road.

We generated three sets of driving scenarios (Fig. 5), invited four human drivers to drive the vehicle manually in those situations, and recorded their demonstrations. The drivers were asked to drive in a way that "would make them feel comfortable as if the car is in autonomous driving mode." The operation was performed under fixed conditions to enforce the same external stimuli to different human drivers, namely the course, object placement, and data-collection vehicle. The idea is to make sure, to the maximum extent possible, that the variance in driving demonstrations was solely caused by the difference in individual driving preferences under the same condition.

Each demonstration is a recorded trajectory with position, heading and speed information. In order to condition the demonstration for model training, we recorded four demonstrations for each scenario/person, and applied dynamic time warping (DTW) [16] to obtain a single smoothed demonstrated trajectory. The data collected in scenarios (S1) and (S2) will be used to show the validity of the proposed planning/learning approach in mimicking human demonstrations. Scenario (S3) is used to evaluate the generative capability of the learned planner.

B. Learning for Individual Preference

The convergence of the proposed learning algorithm in one demonstration is illustrated in Fig. 6 and 7. Applying this algorithm to the collected demonstrations yields Fig. 8.

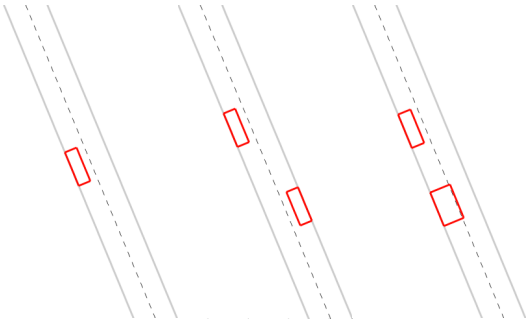


Fig. 5: Three scenarios for human demonstration. The solid gray lines on both sides indicate the lane boundary, and the dashed gray line indicates the centerline of the lane. The leftmost figure shows a single-object scenario (S1). The middle figure shows a two-sided double-object scenario (S2). The rightmost figure shows a single-sided double-object scenario (S3).

A qualitative analysis shows that there are clear differences in driving styles between different people facing the same scenario. Fig. 8a, 8c, 8e illustrate how different people can vary significantly in avoidance maneuvers for each tested scenario, for example: user 1 is the most conservative and tries to slow down for all scenarios even if the path geometry allows driving at a higher speed.

Quantitatively speaking, entries 1-8 in Table I (Fig. 8b and 8d) summarize the detailed parameter learning results, as well as learning accuracies, of a total of 8 DTW-processed driving demonstrations in scenarios (S1) and (S2). The results show an average of 0.12m path error, 0.23m/s speed error for scenario (S1) and 0.22m path error, 0.51m/s speed error for scenario (S2).

These are reasonable learning results due to the limited sampling resolution of the path planner (longitudinal/lateral resolutions are 2.0/0.2m respectively), and the curvature-sensitive nature of the speed profile generation routine. Meanwhile, the optimality criterion design of the planning models only captures the most dominant trade-off aspects, which inevitably imposes limitations on the learning accuracy, which can be observed in Abbeel [13].

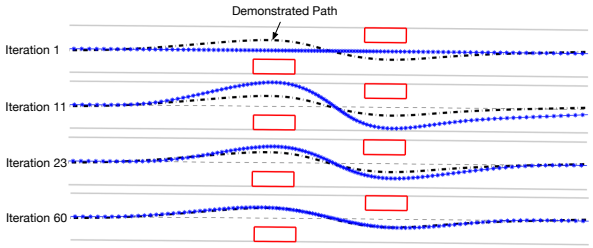


Fig. 6: Learning of one path demonstration of a two-object swerve. The gray lines mark the road, while the red rectangle represents the obstacle. The black dashed curve is the path demonstration, the multiple blue starred curves represent the gradually learned path, and the blue starred curves illustrate the convergence of learning of the demonstrated speed profile.

TABLE I: Summary of Learning Results & Accuracies

Entry #	1	2	3	4	5	6	7	8	9	10	11	12
Scenario #	S1				S2				S3			
User #	1	2	3	4	1	2	3	4	1	2	3	4
ω_{offset}	0.10	0.10	0.03	0.51	0.48	0.19	0.03	0.34	0.10	0.10	0.03	0.51
$\omega_{swerve}^{outside}$	0.56	0.58	0.19	0.19	0.20	0.37	0.38	0.21	0.56	0.58	0.19	0.19
ω_{swerve}^{inside}	0.13	0.10	0.42	0.13	0.14	0.35	0.51	0.34	0.13	0.10	0.42	0.13
ω_{swerve}^{other}	0.01	0.03	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.03	0.01	0.01
$\omega_{obstacle}$	0.20	0.19	0.35	0.16	0.17	0.08	0.07	0.10	0.20	0.19	0.35	0.16
$l_d(m)$	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0
$a_{lat}(m/s^2)$	0.60	1.40	0.07	3.9	0.79	0.78	0.66	0.89	0.60	1.40	1.07	3.9
$a_{lon}^{+}(m/s^2)$	0.80	0.12	0.08	0.13	1.49	1.18	0.60	0.71	0.80	0.12	0.08	0.13
$a_{lon}^{-}(m/s^2)$	0.84	0.01	0.07	0.41	1.42	1.87	1.16	1.57	0.84	0.01	0.07	0.41
$\epsilon_{path}(m)$	0.12	0.09	0.08	0.22	0.25	0.16	0.25	0.23	0.49	0.51	0.34	0.64
$\epsilon_{speed}(m/s)$	0.57	0.08	0.04	0.17	0.57	0.70	0.27	0.52	2.54	5.13	4.91	1.02

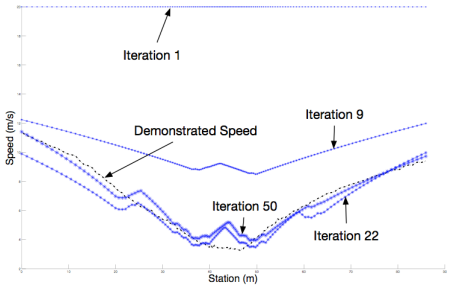


Fig. 7: Learning of one speed demonstration of a two-object swerve. The black dashed curve represents the recorded speed profile, while the blue starred curves illustrate the convergence of learning of the demonstrated speed profile.

C. Generative capabilities of the learned planners

We further applied the learned parameters from scenario (S1) to a new scenario (S3) to evaluate the predictive capability of the learned planner. As illustrated in Fig. 5, (S3) is a setup with two different-width obstacles on the same side. Qualitatively speaking, it is similar to (S1) in that the actual swerve contains only nudging right and getting back to the lane center. Entries 9-12 in Table I (Fig. 8f) show the predictive results. Unfortunately, the prediction results demonstrate a relatively large behavior discrepancy in the test scenario (0.49m path error and 3.4m/s speed error). Our analysis suggests that it is caused by the underlying stochastic human driving nature. More training demonstrations are needed in order to capture a more generative driving pattern.

VI. CONCLUSIONS

In this paper, we presented a novel on-road motion planner for swerve maneuvers. Moreover, we applied inverse reinforcement learning techniques to learn different driving styles from human demonstration. Experimental results show the proposed planner's potential to mimic individual driving style differences. However, due to the limited dataset and parameter over-fitting, difficulties remain in capturing a more generative driving pattern. Future work involves collecting more datasets for further training, possibly along with deep reinforcement learning (Mnih [17]) to enable more generative abstraction.

REFERENCES

[1] J.-W. Lee and B. Litkouhi, "A unified framework of the automated lane centering/changing control for motion smoothness adaptation," in *Intelligent Transportation Systems (ITSC), International IEEE Conference on*, pp. 282–287, IEEE, 2012.

[2] S. Thrun *et al.*, "Stanley: The robot that won the DARPA Grand Challenge," *Journal of Field Robotics*, vol. 23, no. 9, 2006.

[3] C. Urmson *et al.*, "CMU Team: Autonomous driving in urban environments: Boss and the Urban Challenge," *Journal of Field Robotics*, vol. 25, pp. 425–466, Aug. 2008.

[4] J. Ziegler and C. Stiller, "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios," *Intelligent Robots and Systems, International Conference on*, pp. 1879–1884, 2009.

[5] M. McNaughton, C. Urmson, J. M. Dolan, and J.-w. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4889–4895, 2011.

[6] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for Bertha - A local, continuous method," in *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 450–457, 2014.

[7] X. Li, Z. Sun, A. Kurt, and Q. Zhu, "A sampling-based local trajectory planner for autonomous driving along a reference path," pp. 376–381, 2014.

[8] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," *Ann Arbor*, vol. 1001, p. 48105, 2008.

[9] Q. T. Dinh and M. Diehl, "An application of sequential convex programming to time optimal trajectory planning for a car motion," in *Proceedings of the IEEE Conference on Decision and Control*, vol. 0, pp. 4366–4371, IEEE, 2009.

[10] J. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization," in *Robotics: Science and Systems*, vol. 9, pp. 1–10, Citeseer, 2013.

[11] J. Peng, W. Luo, W. Liu, W. Yu, and J. Wang, "A suboptimal and analytical solution to mobile robot trajectory generation amidst moving obstacles," *Autonomous Robots*, vol. 39, pp. 1–23, feb 2015.

[12] B. Hamner, S. Singh, and S. Scherer, "Learning obstacle avoidance parameters from operator behavior," *Journal of field Robotics*, vol. 23, no. 1112, pp. 1037–1058, 2006.

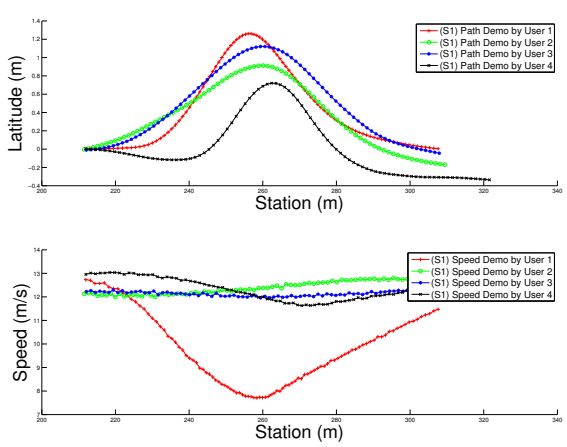
[13] P. Abbeel, *Apprenticeship learning and reinforcement learning with application to robotic control*. ProQuest, 2008.

[14] D. Silver, J. A. Bagnell, and A. Stentz, "Learning from demonstration for autonomous navigation in complex unstructured terrain," *The International Journal of Robotics Research*, 2010.

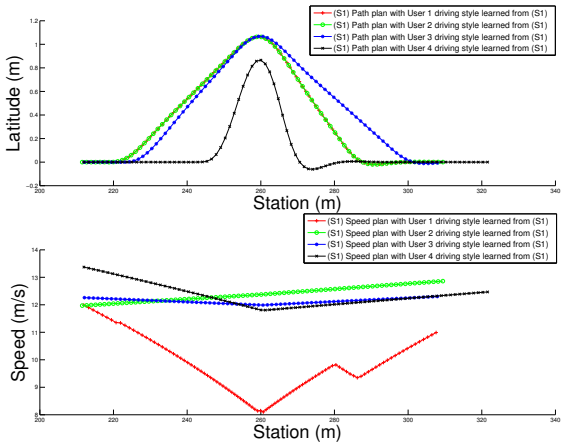
[15] T. Gu and J. M. Dolan, "Toward human-like motion planning in urban environments," in *Intelligent Vehicles Symposium Proceedings*, pp. 350–355, IEEE, 2014.

[16] F. Petitjean, A. Ketterlin, and P. Gançarski, "A global averaging method for dynamic time warping, with applications to clustering," *Pattern Recognition*, vol. 44, pp. 678–693, Mar. 2011.

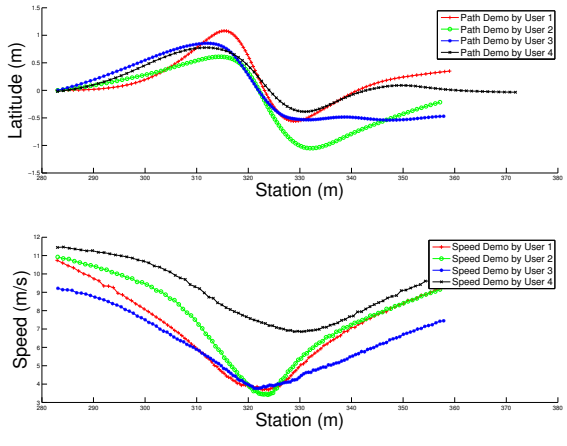
[17] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.



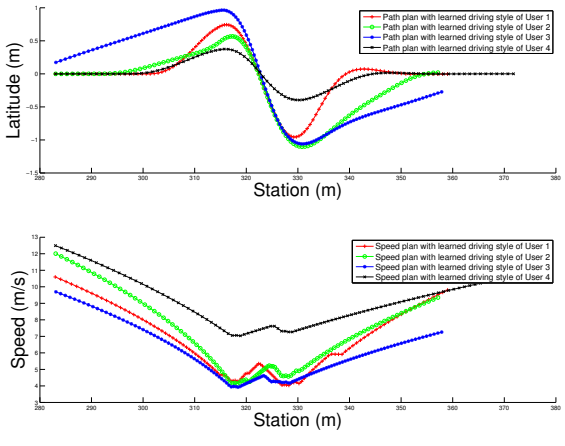
(a)



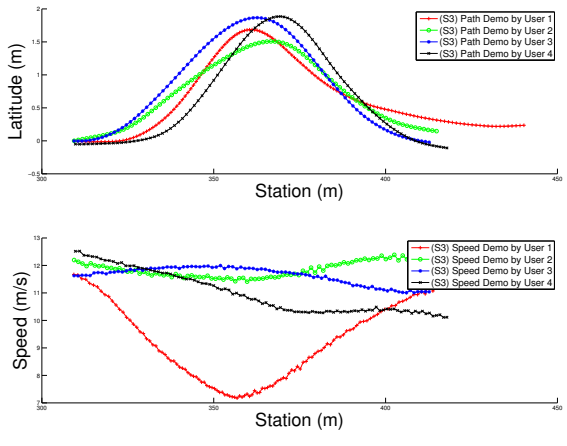
(b)



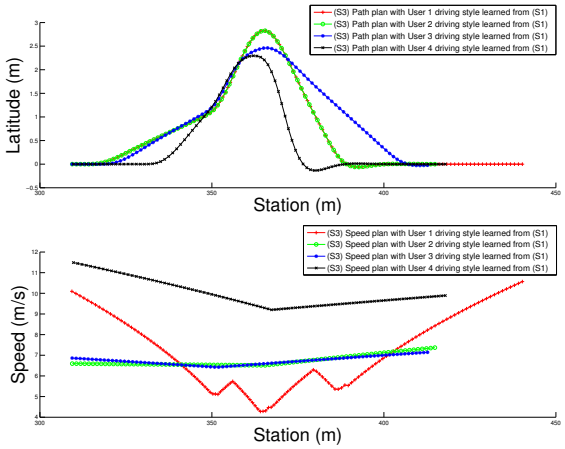
(c)



(d)



(e)



(f)

Fig. 8: Illustrations of user demonstrations, planning results with learned parameters & predictive planning results with learned parameters. (a) (c) (e) shows the user demonstrations in scenario (S1) (S2) (S3) respectively. (b) (d) shows the planning outcomes of parameter-learned planners in the same scenarios, whose demonstration data are used for learning. (f) shows the planning outcomes in scenario (S3) with parameters learned from (S1) demonstrations.